Автоматизация и моделирование в проектировании и управлении. 2025. № 3 (29). С. 54-63.
ISSN 2658-3488 print, ISSN 2658-6436 online
Automation and modeling in design and management. 2025. № 3 (29). P. 54-63.

Управление в организационных системах

Научная статья
Статья в открытом доступе
УДК 65.011.56
doi: 10.30987/2658-6436-2025-3-54-63

РАЗРАБОТКА МОДЕЛИ КОМПЕТЕНЦИЙ ИТ-СПЕЦИАЛИСТОВ НА ОСНОВЕ ИНТЕЛЛЕКТУАЛЬНОГО АНАЛИЗА ДАННЫХ

Эрчимэн Иванович Гаврильев
Новосибирский государственный технический университет, г. Новосибирск, Россия
erchimen_gavriliev@outlook.com, https://orcid.org/ 0000-0001-7289-3969

Аннотация. *Основными факторами, влияющими на результаты ИТ-проектов, являются персонал и его профессиональное развитие. Оценка персонала является ключевым инструментом для определения потенциала сотрудников и возможных карьерных путей. В большинстве случаев оценка основана на субъективных мнениях руководителей, которые не обладают высокой степенью надежности и полноты. Решение этой проблемы видится в использовании результатов анализа цифрового следа. Для анализа были применены методы статического анализа кода и анализа текстов на естественном языке. В данном исследовании была построена матрица компетенций разработчика, на основе результатов факторного анализа и структурного моделирования. Были выявлены компетенций, связанные с разработкой, подходом к решению задач и коммуникациями. Построенная матрица компетенций в дальнейшем может быть использована для оценки компетентности разработчика.*

Ключевые слова: разработчик ПО, модель компетенций, интеллектуальный анализ данных, обработка естественного языка, факторный анализ, структурное моделирование

Для цитирования: Гаврильев Э.И. Разработка модели компетенций ИТ-специалистов на основе интеллектуального анализа данных // Автоматизация и моделирование в проектировании и управлении. 2025. №3 (29). С. 54-63. doi: 10.30987/2658-6436-2025-3-54-63.

Original article
Open Access Article

DEVELOPING AN IT SPECIALIST COMPETENCY MODEL BASED ON INTELLIGENT DATA ANALYSIS

Erchimen I. Gavriliev
Novosibirsk State Technical University, Novosibirsk, Russia
erchimen_gavriliev@outlook.com, https://orcid.org/ 0000-0001-7289-3969

Abstract. *The key factors influencing the outcomes of IT projects are personnel and their professional development. Personnel assessment is a crucial tool for determining employees' potential and possible career paths. In most cases, assessment is based on managers' subjective opinions, which lack high reliability and completeness. Solving this problem is seen in using digital footprint analysis results. For the analysis, the author applies methods of static code analysis and natural language text analysis. This study constructs a developer's competency matrix based on the results of factor analysis and structural modelling; identifies competencies related to development, problem-solving approach, and communication. The constructed competency matrix can be further used to evaluate the developer's competence.*

Keywords: software developer, competency model, intelligent data analysis, natural language processing, factor analysis, structural modelling

For citation: Gavriliev E.I. Developing an IT Specialist Competency Model Based on Intelligent Data Analysis. Automation and modeling in design and management, 2025, no. 3 (29). pp. 54-63. doi: 10.30987/2658-6436-2025-3-54-63.

Введение

ИТ-сфера характеризуется быстрым развитием и постоянной сменой технологий для разработки ПО. Поэтому компаниям важно уделить внимание управлению профессиональным развитием, которое занимается созданием и реализацией программ, направленных на повышение квалификации сотрудников. Основополагающей задачей данного направления является оценка персонала с целью выявления несоответствий между необходимыми знаниями, умениями и навыками (ЗУН) и фактическим уровнем их владения. На основе результатов оценки определяются направления развития.

В ИТ-компаниях чаще всего оценка основывается на субъективных мнениях руководителей и коллег. Этот способ критикуют, указывая на его низкую степень надежности и точности [1]. В существующих работах, изучающих особенности оценки ИТ-специалистов, уделяется внимание техническим компетенциям. Анализ только технических компетенций недостаточен для оценки, так как деятельность ИТ-специалистов диверсифицирована по множеству направлений, в связи с чем требуется учитывать нетехнические (софт) компетенции.

В ряде публикаций рассматриваются методы анализа цифрового следа для оценки квалификации. Под цифровым следом понимаются данные, оставленные ИТ-специалистом при использовании различных информационных систем в процессе работы [2]. Эти данные можно проанализировать при помощи методов интеллектуального анализа данных для оценки квалификации. В связи с чем, целью этой работы является повышение объективности оценки квалификации ИТ-специалистов на основе предложенной модели компетенций.

Обзор литературы

Для управления развитием карьеры в ИТ-организациях часто применяют модель «партнерства по планированию и развитию карьеры», представленную на рис. 1. Данная модель представляет управление профессиональным развитием в виде цикла, в рамках которого руководитель определяет направление профессионального роста, составляет план развития и согласовывает его с сотрудником [3].

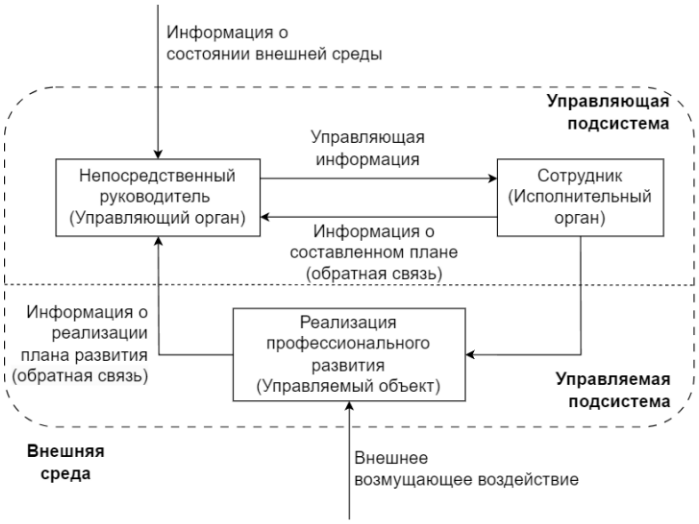

Рис. 1. Обобщенная структурная схема модели «партнерства по планированию и развитию карьеры»
Fig. 1. Generalized diagram of «career planning and development partnership» model

При определении направления профессионального роста важно учитывать потребности организации и сотрудника, которые выявляются в ходе оценки квалификации. Целью оценки является установление степени соответствия между характеристиками сотрудника и требованиями должности. В настоящей работе рассматривается компетентностный подход к оценке, в соответствии с которым учитываются ЗУН сотрудника, а не только его результаты работы, как в традиционном подходе к оценке. Под компетенцией понимается набор ЗУН, которые необходимы для эффективного исполнения должностных обязанностей, а компетентность – готовность применить компетенции [4].

В исследованиях необходимые компетенции разработчиков рассматриваются с точки

зрения их основной деятельности – программирования. Среди необходимых ЗУН часто упоминались знания языков программирования, библиотек и подходов для разработки, навыки администрирования инфраструктуры и технологий для поддержки процесса разработки, например, *Git* [5]. В исследованиях также часто фигурировали навыки проектирования систем, тестирования приложений и анализа исходного кода, отладки [6].

Технические компетенции играют основную роль в квалификации разработчиков, однако большую часть рабочего дня ИТ-специалисты уделяют другим активностям: поиску информации, планированию, менторству и т.д. [7]. Следовательно, такие софт компетенции, как коммуникации и работа в команде, приобретают важное значение. Актуальные исследования предоставляют комплексное видение компетенций разработчиков, но они не предлагают показатели, которые можно использовать для оценки квалификации сотрудника [8]. Предложенные методы оценки больше подходят для кандидатов на работу, чем для уже работающих сотрудников, поскольку они не учитывают вклад ИТ-специалистов [9].

Разработка ПО генерирует большие объемы данных в различных информационных системах, которые могут помочь в управлении развитием персонала. Область деятельности по извлечению и анализу информации из этих систем получила название «анализ репозиториев ПО» (*Mining Software Repository*, *MSR*) [10]. Исследования в области *MSR* утверждают, что специалист, внесший значительный вклад в репозиторий проекта, может обладать более высоким уровнем компетентности [11].

На основании обзора литературы можно заключить, что метод оценки, основанный на анализе данных из систем, обладает более высокой степенью объективности по сравнению с мнениями руководителей, поскольку он опирается на количественно измеримые показатели и актуальные данные. Однако в этих исследованиях квалификация разработчиков рассматривалась в контексте разработки ПО с открытым исходным кодом, что отличается от условий разработки проприетарного ПО. Поэтому необходимо провести исследование, направленное на изучение применимости метода для оценки квалификации разработчиков в данных условиях.

Сбор и обработка данных для оценки квалификации разработчиков

В исследованиях распространенным источником данных для анализа цифрового следа является система контроля версий (СКВ), которая обеспечивает хранение и отслеживание изменений исходного кода. Однако трудовые обязательства разработчиков ограничиваются не только программированием, сотрудники также активно участвуют в выполнении задач, связанных с нетехническими компетенциями. Поэтому требуется провести анализ сведений из информационных систем, поддерживающих такие активности – система управления задачами (СУЗ) и система управления знаниями (СУЗн).

Взаимодействие в рамках разработки (регистрация задач и ошибок, мониторинг их решения и т.д.) осуществляется с помощью СУЗ, предназначенной для организации работы проектных команд. В контексте СУЗ задача представляет собой задание, которое требует от исполнителя выполнения определенных действий. Задачи могут варьироваться по приоритету, типу и сроку.

Система управления знаниями играет важную роль в хранении и обмене информации, используемой при разработке и сопровождении ПО. В СУЗн вся информация хранится на страницах, которые может создать и отредактировать сотрудник.

Из описанных источников загружаются только те данные, которые могут охарактеризовать то, как разработчик исполняют трудовые обязанности. Из СКВ загружается содержимое репозиториев: исходный код приложений и коммиты (снимки состояний исходного кода). Из СУЗ загружаются задачи, комментарии к ним, записи из журнала работ и история изменений задач. Из СУЗн выгружаются страницы, история их изменений и комментарии к ним.

Информация, полученная из систем, подвергается анализу для вычисления показателей, которые впоследствии могут служить основой для оценки квалификации сотрудников. Исследование применимости данных для оценки было проведено в компании среднего размера, которая занимается разработкой ПО для банковской отрасли. Основным инструментом для создания программных продуктов является их собственный конструктор приложений.

В зависимости от формы данных различаются способы их обработки: структурирован-

ные – данные, организованные в заранее определенные форматы, например, числовые значения, даты и систематизированные текстовые строки; неструктурированные – данные, характеризующиеся отсутствием четкой организации.

К структурированным данным, собранным из источников, относятся:

– метаданные коммитов – список измененных объектов;

– метаданные задач, комментариев и журнала работ из СУЗ – автор, исполнитель, приоритет, статус, объем времени, выделенный на задачу;

– метаданные страниц из СУЗн – автор, пространство, количество отметок «нравится».

Эти метаданные используются для расчета показателей квалификации. Например, на основе данных из СКВ рассчитываются: среднее количество коммитов в день; количество типов объектов, разработанных в конструкторе и т.д. На основе данных из СУЗ рассчитываются: количество задач, решенных в рамках и вне рамок оценки; количество назначенных задач в разрезе приоритетов и т.д. Сведения из СУЗн используются для расчета показателей активности сотрудника в процессах передачи знаний: количество созданных страниц; количество обновлений содержимого страниц и т.д.

К неструктурированным данным относятся: исходный код, содержимое страниц из СУЗн, названия и описания задач, записей из журнала учета работ, а также содержимое самих комментариев.

Для расчета показателей на основе неструктурированных данных необходимо проводить операции по преобразованию и извлечению полезной информации. Наиболее часто в исследованиях применяются методы статического анализа кода для оценки технических компетенций. Например, рассчитываются метрики сложности кода, т.к. считается, что квалифицированные разработчики могут писать более читаемый код с низкой степенью сложности. В состав показателей квалификации были включены: цикломатическая сложность МакКейба, метрики Холстеда и поддерживаемость кода. Формула расчета цикломатической сложности представлена ниже [12]:

$$Z(G) = l - v + 2 \cdot p,$$

где l – число дуг ориентированного графа потока управления кода G; v – число вершин; p – число компонентов связности графа.

Метрики статического анализа кода рассчитывается для всех скриптов, написанных разработчиками, а при оценке квалификации рассчитывается среднее значения метрик. Кроме того, скрипты также обрабатываются для подсчета применяемых элементов языка программирования: количество применений *Java*-библиотек, функций для интеграции с внешними системами и т.д. Дополнительно проводится анализ составленных *SQL*-запросов для расчета показателей подсчета применяемых элементов *SQL*.

Часто сотрудники ведут подробную отчетность на естественном языке (ЕЯ) о результатах своей работы в СУЗ. Эта информация может быть использована для оценки квалификации при помощи инструментов и подходов анализа текстов на ЕЯ. В рамках текущей работы используется подход, основанный на правилах, который задействует предопределенные шаблоны для обработки текстов.

Для выявления фактов – утверждений, представляющих конкретные события – требуется подготовить лексико-синтаксические паттерны (ЛСП). Лексико-синтаксические паттерны – шаблонные сочетания слов, которые следуют определенным грамматическим и семантическим правилам. В ЛСП для извлечения фактов применяются предметные словари, содержащие термины для поиска слов.

Для оценки квалификации предлагается анализировать тексты на ЕЯ из СУЗ, чтобы выявить факты применения технических средств и инструментов для разработки, а также типов решенных задач. Перед извлечением этой информации требуется провести анализ, целью которого является составление ЛСП, в соответствии с шагами ниже.

На первом этапе были составлены базовые ЛСП, которые выявляют согласованные фразы, в которых может содержаться искомая полезная информация. Эти ЛСП были составлены на основе обзора литературы и предварительного анализа данных. При помощи них извлекаются фразы в форме «глагол + существительное (объект действий)», например, «проанализировать требования», «исправить дефект» и т.д.

На втором этапе для извлечения фраз были обработаны сведения из СУЗ в период с 2021

по 2022 год. Перед проведением анализа из них исключались стоп-слова, а также гиперссылки. Всего было выявлено более 19 тысяч фраз.

На третьем этапе были проанализированы извлеченные фразы. Они были распределены по группам в зависимости от объекта действий, например, «форма», «демонстрация», «инструкция» и т.д. Данное распределение позволило выделить тематические категории фактов. Например, фразы «написать инструкцию», «добавлять в инструкцию» относятся к составлению документации. При этом следует отметить, что некоторые объекты действия относятся к общей области знаний. Например, фразы, относящиеся к разработке формы и настройке отдельного виджета, объединены в один тип задачи – разработка интерфейса.

На последнем этапе составлялись новые ЛСП и предметные словари для извлечения фактов. Новые ЛСП основывались на структуре базовых ЛСП из первого этапа. Для поиска ключевых слов использовались предметные словари, которые были составлены при помощи выявленных терминов из третьего этапа, а также общей терминологии ИТ-области. Всего было составлено 74 ЛСП. При оценке квалификации разработчика рассчитывалось количество найденных фактов за период оценки. Для анализа текста был применен Томита-парсер, извлекающий факты из текстов на русском языке при помощи ЛСП.

Моделирование

Первичная система показателей квалификации включала в себя более 250 показателей. Для изучения применимости сформированной системы показателей для оценки квалификации был проведен анализ данных.

Для начала был проведен сбор оценок квалификации ИТ-специалистов от руководителей. Начальная версия выборки включала в себя 651-ну оценку 153-х разработчиков в период с 2019 по 2024 гг. Руководители выставляют оценку по восьми балльной шкале.

Для моделирования оценки были задействованы методы многомерного анализа данных, в качестве входных данных используются показатели квалификации, а в качестве зависимой переменной выступала оценка руководителя (*prof*).

По результатам разведочного анализа из выборки были исключены ИТ-специалисты, проработавшие в компании меньше трех месяцев, так как для них отсутствует значительная часть информации. Также из выборки были исключены руководители команд, поскольку они чаще всего занимаются менеджментом. Кроме того, были удалены показатели, у которых отсутствовали значения для всех сотрудников. После корректировок выборка включала в себя 541-ну оценку и 206 показателей.

Для структурирования пространства показателей квалификации был применен метод главных компонент (*Principal Component Analysis*, *PCA*). Для выборки разработчиков критерий Бартлетта давал уверенные показания для проведения анализа – 79 564, *p*-значение $< 2e\text{-}16$.

Для выбора количества факторов учитывался график каменистой осыпи. В соответствии с графиком, представленном на рис. 2, значение равно 31.

Для расчета нагрузок факторов был применен метод *varimax*, абсолютные значения меньше 0,4 не были включены в факторную модель. Для проверки соответствия модели данным был проведен подтверждающий факторный анализ (*Confirmatory Factor Analysis*, *CFA*). В результате проведения анализа состав факторов был изменен, из него были исключены избыточные факторы, а итоговая модель состояла из 19-ти факторов. Полученные результаты *CFA* свидетельствуют об относительно удовлетворительном качестве модели: *CFI* (сравнительный индекс соответствия, $> 0{,}950$) – 0,610, *RMSEA* (среднеквадратичная ошибка аппроксимации, $< 0{,}1$) – 0,076, *SRMR* (стандартизированный среднеквадратичный остаток, $< 0{,}1$) – 0,074. Построенная модель включает в себя факторы, представленные ниже.

*RC*1 «Задачи». Данный фактор позволяет оценить сложность и качество работы разработчика с точки зрения решенных задач. Он включает в себя следующие показатели: количество назначенных задач с приоритетами «критический» и «средний»; среднее количество комментариев за один день; общее количество комментариев; среднее количество времени, выделенное на решение одной задачи; количество исправленных дополнительных ошибок, решенных в рамках одного дефекта; количество решенных задач; количество решенных задач в рамках оценки; количество переоткрытий задач.

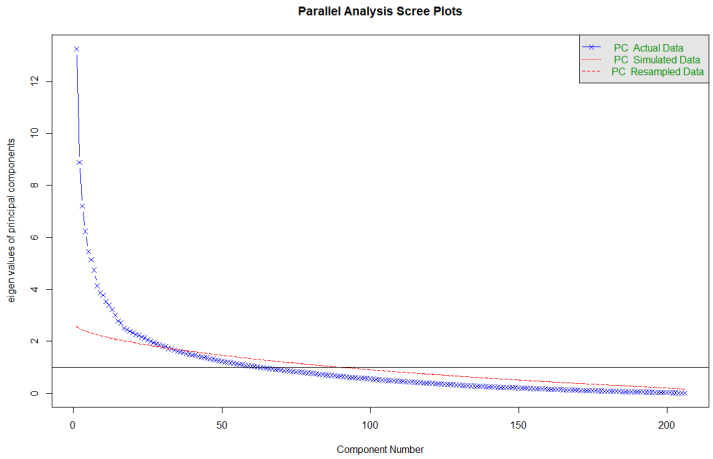

Рис. 2. График «Каменистая осыпь»
Fig. 2. Scree plot

*RC*2 «Взаимодействие с внешними субъектами». Фактор характеризует то, как разработчик обрабатывает задачи, зарегистрированные клиентами и партнерами. Все показатели этого фактора рассчитываются на основе задач, зарегистрированных в системе управления задачами для внешних субъектов: количество назначенных задач с приоритетами «блокирующий», «критический», «средний» и «незначительный»; среднее количество комментариев за 1 день; среднее количество изменений статусов назначенных задач; общее количество комментариев задачах; колличество решенных задач от внешних субъектов.

*RC*3 «Работа с СУБД». Этот фактор охватывает аспекты разработки элементов приложения, взаимодействующих с СУБД, он включает в себя такие показатели как: количество созданных индексов; количество поисковых методов с типом «конструктор» / «*SQL*»; среднее количество составленных *SQL*-запросов с разными инструкциями; количество разработанных пользовательских функций и бизнес-правил; количество назначенных задач с приоритетом «незначительный».

*RC*4 «Моделирование бизнес-процессов». Охватывает то, как разработчик моделирует бизнес-процессы (БП) в конструкторе. Фактор включает в себя показатели подсчета элементов нотации *BPMN*: количество блоков с типами «вызов подпроцесса», «исключающее ИЛИ»; количество переходов с разными типами.

*RC*5 «Работа с архитектурой системы». Описывает взаимодействие разработчика с архитектурными компонентами системы. Включает в себя следующие показатели: количество созданных *REST*-сервисов, параметров окружения, *JavaScript*-функций и библиотек, модулей приложений и перечислений, а также количество зарегистрированных задач по поддержке конструктора приложений и частота упоминаний планирования задач.

*RC*6 «Участие в распространении знаний». Включает показатели, связанные с обменом знаниями, такие как: общее количество созданных страниц в СУЗн; количество обновлений содержимого страниц; количество созданных страниц в пространстве по обучению конструктору; частота упоминаний составления планов развития и работы с системой мониторинга.

*RC*7 «Администрирование инфраструктуры». Охватывает аспекты обслуживания и управления серверными окружениями. Фактор включает в себя такие показатели, как: частота упоминаний команд по администрированию сервера приложений и СУБД, *Java*, команд диагностики – *top*, *jstack*, *jmap*, а также обновлений конструктора приложений.

*RC*8 «Базовые навыки разработки». Содержит показатели, свидетельствующие о наличии базовых навыков разработки: частота упоминаний разработки БП, скриптов, форм, модели данных, работы с документацией, тестирования и разработки прототипов.

*RC*9 «Настройка фоновых задач». Связан с разработкой фоновых процессов и задач в приложениях. Состоит из показателей: количество настроенных блоков в БП с типом «таймер», «обработка ошибок» и «оператор И», а также частота упоминаний администрирования задач.

*RC*10 «Написание функций модуля». Включает в себя показатели подсчета созданных функций модуля и редактирований функций модуля, изначально разработанных другими сотрудниками, а также количество блоков БП с типом «отправка электронного письма».

*RC*11 «Поддерживаемость и сложность написанного кода». Состоит из следующих показателей: средние значения метрик статического анализа кода, написанного разработчиком, как цикломатическая сложность, сложность, объем и словарь программы по Холстеду, а также уровень поддерживаемости кода.

*RC*12 «Постановка задач. Описывает процесс формулирования и управления задачами. Включает в себя показатели: частота упоминаний консультирования, анализа и оценки требований, постановки задач, контроля статуса задач и среднее количество задач, по которым работает сотрудник в один день.

*RC*13 «Активность решения задач». Характеризует продуктивность разработчика в решении задач. В фактор включены показатели: среднее количество решенных задач за один день, количество изменений статусов задач за один день; количество загруженных *Java*-компонент.

*RC*14 «Отладка. Охватывает аспекты отладки: частота упоминаний решения проблем, отладки и исправления ошибок, а также обучения заказчиков; среднее количество применения функций по взаимодействию с внешними системами через *SOAP*-протокол.

*RC*15 «Наставничество». Связан с передачей знаний и наставничеством. Фактор включает в себя: количество обученных стажеров; количество участий в консультациях стажеров; количество разработанных БП и количество редактирований БП, разработанных другими сотрудниками; количество загруженных тем приложений.

*RC*16 «Разработка экранных форм». Описывает ЗУН разработки пользовательских интерфейсов. Фактор состоит из показателей: количество разработанных форм; количество редактирований форм, разработанных другими сотрудниками; количество разработанных абстрактных форм.

*RC*17 «Оптимизация». Содержит переменные, связанные с увеличением производительности приложений. Фактор включает в себя: частота упоминаний профилирований *SQL*-запросов, оптимизации приложений, изменений поисковых методов и печатных форм, а также количество разработанных поисковых методов с типом «полнотекстовый поиск».

*RC*18 «*SOAP*-сервисы». Охватывает аспекты работы с *SOAP* веб-сервисами. Состоит из частоты упоминаний редактирования *WSDL*-схем.

*RC*19 «Настройка обработчиков событий экранных форм». Описывает то, как разработчик настраивает обработчики событий в пользовательских интерфейсах. Состоит из таких показателей как: количество переходов между операторами в обработчиках событий с условиями и без условий; количество разработанных скриптов для взаимодействия с серверами и скриптов для изменения пользовательского интерфейса.

Для выявления ключевых факторов, влияющих на оценку руководителя, применен метод структурного моделирования (*Structural Equation Modeling*, *SEM*), использующий результаты применения метода PCA. Построенная структурная модель, представленная ниже, состоит из девяти факторов, из нее были исключен незначимые факторы:

$$prof = 2{,}434 \cdot RC2 + 5{,}329 \cdot RC5 + 3{,}304 \cdot RC6 + 1{,}146 \cdot RC12 +$$
$$+ 1{,}963 \cdot RC13 + 2{,}814 \cdot RC14 + 2{,}831 \cdot RC15 - 3{,}230 \cdot RC16 - 0{,}991 \cdot RC18.$$

На рис. 3 представлена структурная модель. Показатели *CFA* структурной модели свидетельствуют об относительно приемлемом качестве: *CFI* – 0,659; *RMSEA* – 0,097; *SRMR* – 0,083.

Построенную модель можно интерпретировать следующим образом: положительно на оценку квалификации от руководителя влияет фактор *RC*2, связанный со взаимодействием с внешними субъектами. Наличие развитых коммуникативных компетенций позволяют разработчикам эффективно передавать информацию о текущем статусе выполнения задач. Кроме того, разработка проприетарного ПО часто требует адаптации к изменяющимся условиям и требованиям клиентов. Умение слушать и задавать вопросы помогает разработчикам быстро реагировать на изменения и вносить необходимые корректировки.

Положительно на оценку влияют ЗУН, связанные с распространением знаний и наставничеством (*RC*6, *RC*15). Распространение знаний помогает разработчикам лучше понимать свои области экспертизы и выявлять возможные пробелы в своих знаниях. Также наставничество может улучшить качество работы проектной команды и способствовать профессиональному росту самого ментора и его коллег.

Фактор *RC*12, связанный с постановкой и администрирования задач, также положительно влияет на оценку. Умение формулировать задачи для других членов команды позволяет разработчику делегировать работу, а также оперативнее достигать целей самого проекта. Контроль исполнения задач проектной команды позволяет разработчику выявлять возможные проблемы на ранних стадиях, что снижает риски задержек. При оценке руководители учитывают оперативность решения задачи при оценке разработчика (*RC*13), так как этот фактор тесно связан с успешностью проекта – сроком и бюджетом. Чем дольше решается задача, тем выше становится уровень операционных расходов.

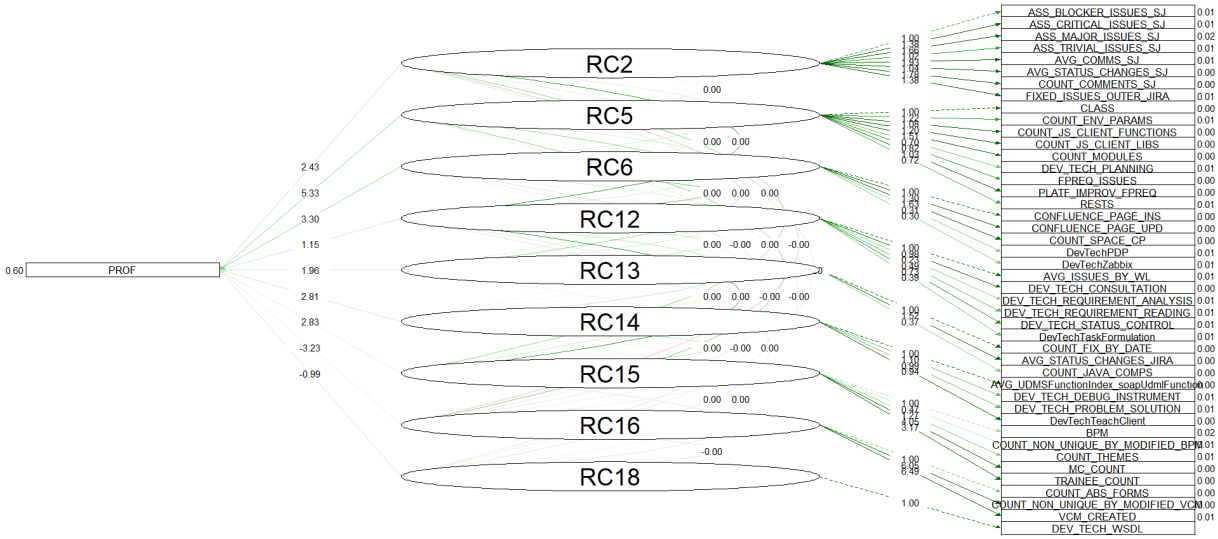

Рис. 3. Структурная модель зависимости оценки квалификации разработчика
Fig. 3. Structural model of the dependency of a developer's qualification assessment

Основной чертой компетентности разработчика являются глубокие знания инструментов и способов их применения, позволяющие снизить трудозатраты на решение задач, поэтому положительное влияние на оценку оказывает фактор работы с архитектурой системы (*RC*5). Отладка – это неотъемлемая часть процесса разработки ПО. Умение выявлять и устранять ошибки в коде позволяет разработчикам обеспечивать стабильность и надежность приложений. Поэтому положительное влияние на оценку также оказывает фактор, связанный с отладкой (*RC*14).

Однако стоит отметить, что факторы, связанные с разработкой экранных форм (*RC*16) и работой с *WSDL*-схемами (*RC*18), могут негативно сказываться на оценку разработчика. Возможно, в выборке задачи такого типа назначаются на менее опытных сотрудников, так как они не требуют высокого уровня квалификации.

Разработка матрицы компетенций

Основой компетентностного подхода к оценке является модель компетенций, которая определяет, какие ЗУН должны продемонстрировать сотрудники для подтверждения необходимого уровня квалификации. Описание компетенции в модели определяется двумя понятиями – индикатор поведения и уровень владения. Индикатор поведения – стандарт поведения, отражающий действия сотрудника, который обладает соответствующей компетенцией. Для структурирования градации квалификации сотрудника индикаторы поведения объединяются в уровни владения, характеризующие степень развития компетенции, например, «начинающий», «опытный» и «эксперт». Уровни владения представляют собой кумулятивную структуру, в которой сотрудник, обладающий компетенцией на высшем уровне, располагает всем комплексом необходимых ЗУН, присущих предшествующим уровням.

Значимые факторы, выявленные в структурной модели, сформулированы как отдельные компетенции. Однако при оценке руководителями не всегда учитываются все необходимые показатели. Например, это показатели, характеризирующие ЗУН в таких направлениях деятельности, как изучение требований, обучение сотрудников и т.д. Поэтому на основе построенной факторной модели была разработана модель компетенций в виде матрицы.

Строки матрицы – компетенции, а столбцы – уровни владения компетенциями, представленные в виде пяти уровней распространенных квалификаций в ИТ-области, начальным уровнем является «*Trainee*» (стажер), а последним – «*Lead*». В ячейках указываются индикаторы поведения для уровня владения. Фрагмент матрицы компетенций представлен в табл. 1.

Таблица 1

Фрагмент матрицы компетенций

Table 1

Fragment of competency matrix

Trainee	*Junior*	*Middle*	*Senior*	*Lead*
Компетенция: Средства отладки				
Использует для отладки кода вывод информации в консоль или логи.	Пользуется в работе одним из базовых средств отладки. Знает, в каком именно звене могут возникать ошибки.	Умеет пользоваться всеми средствами диагностики и отладки. Умеет собирать полную диагностическую информацию о проблеме.	Умеет интерпретировать текст большинства ошибок, а также собирать информацию о состоянии сервера.	см. *Senior*
Компетенция: *REST*-сервисы				
Н/Д	Умеет настраивать внешние *REST*-сервисы.	Умеет настраивать локальные и внешние *REST*-сервисы.	Может составить документацию для *REST*-сервисов в нотации *openAPI*.	см. *Senior*

Для оценки актуальности матрицы компетентности был проведен опрос 10 разработчиков квалификации «*Senior*» и «*Lead*». В рамках опроса ИТ-специалистам требовалось ознакомиться с матрицей, предоставить обратную связь о ней в свободной форме по указанным вопросам. Респондентам требовалось указать недостающие компетенции, компетенции с нечеткими описаниями и избыточные компетенции, а также обосновать свой выбор. Последняя группа вопросов была направлена на получение общей обратной связи от сотрудников.

Среди списка отсутствующих компетенций часто упоминались компетенции, связанные с управлением команды, включая разрешение конфликтов, планирование и подготовку отчетов. ИТ-специалисты, которые недавно перешли на руководящие роли, могут столкнуться с трудностями в управлении проектными командами из-за большого объема литературы по управленческим практикам и ее применимости к разработке ПО. Таким образом, им требуется руководство по управленческим задачам, которые можно указать в матрице компетенций.

Было отмечено, что продвинутые компетенции, связанные с администрированием ОС *Linux*, не являются обязательными для разработчиков. Разработчикам не обязательно иметь развитые компетенции администрирования, так как появление и развитие области *DevOps* за последнее десятилетие было сосредоточено на эту область деятельности.

Общее мнение участников опроса варьировалось: восемь разработчиков нашли матрицу удобной в использовании и сочли компетенции всеобъемлющими, полагая, что она выполняет свою основную задачу – помогает ознакомиться с необходимыми ЗУН, а также облегчает управление профессиональным развитием. Два разработчика упомянули о трудностях восприятия матрицы из-за большого объема информации.

Разработанная матрица компетенций может быть использована в других ИТ-компаниях, поскольку она охватывает общие профессиональные компетенции. Однако технические компетенции могут не иметь универсального применения, так как требуемые ЗУН могут значительно различаться из-за применяемого технологического стека. Тем не менее, эта модель может подойти для компаний, занимающихся разработкой корпоративных систем, поскольку эти системы часто основываются на аналогичной трехзвенной архитектуре.

В рамках дальнейших работ оценка квалификации будет проводиться при помощи методов нечеткой логики, учитывающей специфику приблизительной оценки. Оцениваться будет степень соответствия индикаторам поведения. Дополнительно планируется провести опрос большего количества разработчиков для валидации матрицы компетенций.

Заключение

При помощи разработанной матрицы компетенций менеджеры могут проводить оценку квалификации разработчика, а разработчики могут определить список компетенций на даль-

нейшее развитие. В рамках будущих работ планируется подготовить модель оценки квалификации, использующей результаты анализа цифрового следа, а также провести опрос на большей выборке.

Список источников:

1. Rothstein H.R. Interrater reliability of job performance ratings: Growth to asymptote level with increasing opportunity to observe // Journal of Applied Psychology. – 1990. – №3 (75). – P. 322-327.
2. Munister V., Zolkin A., Malikov V., Kosnikova O., Poskryakov I. Computational analysis of the digital footprint using machine learning and artificial intelligence // Journal of Physics: Conference Series. – 2021. – №3 (2094). – P. 1-7.
3. Кабанченко Т.С. Психология в управлении человеческими ресурсами. – СПб.: Питер, 2003. – 400 с.
4. Bartram D., Robertson I.T., Callinan, M. Introduction: A Framework for Examining Organizational Effectiveness. – 2002. – 281 p.
5. Surakka S. What subjects and skills are important for software developers? // Communications of the ACM. – 2007. – №1 (50). – P. 73-80.
6. Ahmadzadeh M., Elliman D., Higgins C. An analysis of patterns of debugging among novice // SIGCSE. – 2005. – P. 84-88.
7. Sedelmaier Y., Landes D. Software Engineering Body of Skills (SWEBOS) // 2014 IEEE Global Engineering Education Conference (EDUCON). – 2014. – P. 395-401.
8. Baltes S., Diehl S. Towards a theory of software development expertise // Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE. – 2018.
9. Siegmund J., Kästner C., Liebig J. Measuring and modeling programming experience // Empirical Software Engineering. – 2014. – № 19. – P.1299-1334.
10. Siddiqui T., Ahmad A. Data Mining Tools and Techniques for Mining Software Repositories: A Systematic Review // Big Data Analytics. Advances in Intelligent Systems and Computing. – 2018. – № 654. – P. 717-726.
11. Gousios G., Kalliamvakou E., Spinellis D. Measuring developer contribution from software repository data // Proceedings of the 2008 international working conference on Mining software repositories, MSR ’08. – 2008. – P.129-132.
12. Teusner R., Matthies C., Giese P. Should I Bug You? Identifying Domain Experts in Software Projects Using Code Complexity Metrics // 2017 IEEE International Conference on Software Quality, Reliability and Security (QRS). – 2017.

References:

1. Rothstein H.R. Interrater Reliability of Job Performance Ratings: Growth to Asymptote Level with Increasing Opportunity to Observe. Journal of Applied Psychology. 1990;3(75):322-327.
2. Munister V., Zolkin A., Malikov V., Kosnikova O., Poskryakov I. Computational Analysis of the Digital Footprint Using Machine Learning and Artificial Intelligence. Journal of Physics: Conference Series. 2021;3(2094):1-7.
3. Kabanchenko T.S. Psychology in Human Resource Management. St. Petersburg: Piter; 2003.
4. Bartram D., Robertson I.T., Callinan, M. Introduction: A Framework for Examining Organizational Effectiveness; 2002.
5. Surakka S. What Subjects and Skills are Important for Software Developers? Communications of the ACM. 2007;1(50):73-80.
6. Ahmadzadeh M., Elliman D., Higgins C. An Analysis of Patterns of Debugging Among Novice. SIGCSE. 2005:84-88.
7. Sedelmaier Y, Landes D. Software Engineering Body of Skills (SWEBOS). In: Proceedings of 2014 IEEE Global Engineering Education Conference (EDUCON); Istanbul (Turkey): 2014. p. 395-401.
8. Baltes S, Diehl S. Towards a Theory of Software Development Expertise. In: Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE: 2018.
9. Siegmund J., Kästner C., Liebig J. Measuring and Modelling Programming Experience. Empirical Software Engineering. 2014;19:1299-1334.
10. Siddiqui T., Ahmad A. Data Mining Tools and Techniques for Mining Software Repositories: A Systematic Review. Big Data Analytics. Advances in Intelligent Systems and Computing. 2018;654:717-726.
11. Gousios G, Kalliamvakou E, Spinellis D. Measuring Developer Contribution from Software Repository Data. In: Proceedings of the 2008 International Working Conference on Mining Software Repositories, MSR ’08: 2008. p. 129-132.
12. Teusner R, Matthies C, Giese P. Should I Bug You? Identifying Domain Experts in Software Projects Using Code Complexity Metrics, In: Proceedings of 2017 IEEE International Conference on Software Quality, Reliability and Security (QRS); Prague (Czech Republic): 2017.

Информация об авторах:

Гаврильев Эрчимэн Иванович
аспирант Новосибирского государственного технического университета, ORCID 0000-0001-7289-3969

Information about the authors:

Gavriliev Erchimen Ivanovich
Postgraduate student of Novosibirsk State Technical University, ORCID: 0000-0001-7289-3969

Статья поступила в редакцию 21.01.2025; одобрена после рецензирования 18.02.2025; принята к публикации 18.03.2025.

The article was submitted 21.01.2025; approved after reviewing 18.02.2025; accepted for publication 18.03.2025.

Рецензент – Малаханова А.Г., кандидат технических наук, доцент, Брянский государственный технический университет.

Reviewer – Malakhanova A.G., Candidate of Technical Sciences, Associate Professor, Bryansk State Technical University.